Trustability Improvement of an Automatic Train Protection System

Jing Liu¹¹, Xijiao Xiong¹, Dehui Du¹, Zuohua Ding², Geguang Pu¹

¹ Shanghai Key Lab of Trustworthy Computing, East China Normal University, Shanghai, 200062, P.R.China
jlui@sei.ecnu.edu.cn, yuyaoxi@gmail.com, dhd@gppu1@sei.ecnu.edu.cn
² Center of Math Computing and Software Engineering, Zhejiang Sci-Tech University, Hangzhou, 310018, P.R.China
Zuohuangding@hotmail.com

Abstract. The trustability of control software used in automatic train protection system is crucial to ensure safety of the system. Model-driven paradigm is becoming a de-facto industrial standard for modeling and designing safety critical control systems. This has led to an increased emphasis on setting up a mechanism that can be used to guarantee the correctness of the models. In this paper, we report on a successful application of the experience with Automatic Train Protection system (ATP) modeling techniques. The ATP models are constructed and refined based on Refinement Calculus of Object Systems (rCOS) in Model Driven Architecture (MDA). The time critical interaction mechanism of components is modeled by our extended timed automata.

Keywords: formal method, rCOS, automatic train control, Timed automata.

1 Introduction

With the rapid improvement and innovation of automatic control systems, the complexity of many control systems, especially safety critical systems requires the application of a battery of such techniques. One of the most promising approaches is model driven development.

Up to date, the most advanced train control system is the Communication Based Train Control (CBTC) system [1]. Its rapid deployment has enabled the primary safety features that can be gained from the Automatic Train Protection (ATP) to be introduced for the most safety critical areas [2, 3]. Thus how to model and verify the system to find the design defects has become one of the key issues of ATP [4].

Our approach is sketched as follows. Firstly, an interface model of component is presented with respect to Component-Based and Model-Driven Development specification. The models are defined based on the Refinement Calculus of Object Systems (rCOS) [5, 6]. Secondly, a language-independent dynamic behavior model is constructed by adding the concept of multiple-time to timed automata. In Interface Automata, we introduce the concept of clocks and extended time constraints of timed

¹ Corresponding author
automata. These automata can model the real-time interaction process directly. We apply this model to analysis of the timed behavior of the ATP system.

The paper is organized as follows. Section 2 presents the architecture model of the component-based system by rCOS. Section 3 elaborates on real-time analysis based on component model architecture. Section 4 states the application of the approach to our ATP system. Finally, in Section 5, we conclude our work.

2 Specification and Implementation Models

rCOS is developed to support model-driven development methods. It provides with multi-view modeling environment and combines object-oriented and component-based design and analysis techniques [7].

\[
\begin{align*}
\text{Class Model} & \quad \text{In } rCOS, \text{ a class can be specified:} \\
\text{class} & \quad C \text{ [extends } D] \\
\text{invariant} & \quad \text{Inv} \\
\text{attr} & \quad T_1 x_1 = d_1, \ldots, T_k x_k = d_k \\
\text{method} & \quad m(T \text{in}, V \text{return}) \\
\text{pre:} & \quad p \lor \ldots \lor p \\
\text{post:} & \quad \land (R_1, \ldots; R) \lor \ldots \lor (R; \ldots; R) \\
& \quad \land \ldots \\
\text{method} & \quad m(T \text{in}, V \text{return}) \\
\end{align*}
\]

3 Interface Automata of Components

To describe the timing features of real-time system, we extend timed automata to enhance the ability of modeling multi-clock system. Furthermore, we use an automaton to represent the interaction of components with its environment.

3.1 Time Model

Definition 3.1 [Clock] A clock C is a tuple \( \langle I, L, M, \prec, U \rangle \), where

- I is a set of finite or infinite instants,
- L is a set of labels,
- \( M = I \rightarrow L \) is a mapping that associates labels to instance.
- \( \prec \) is a total, irreflexive and transitive binary relation on I, named precedence.
- U represents a unit.

To express the multiple time features, we extend the clock constraints.

Definition 3.2 [clock constraints] For a set C of clock variables, the set \( \Phi(C) \) of clock constraints \( \delta \) is defined inductively by

\[
\delta := c \leq m \mid c \geq m \mid C_i \preceq C_j \mid C_i \equiv C_j \mid C_i \prec C_j \mid C_i \sim C_j \mid \delta_1 \land \delta_2 \mid \neg \delta
\]

- \( c \leq m \) has the original meaning in timed automata, which means the time of clock \( c \) is less than or equal to \( m \).
• $c \geq m$ means the time of clock $c$ is more than or equal to $m$;
• $\preceq$ (precedence) is a binary relation on $\bigcup_{c \in C} I_c$, which is reflexive and transitive. Formally, let $A$ and $B$ be two discrete-time clocks. A precedence $B$ (denoted $A \preceq B$) if $(\forall i \in I_B) (k = idx_B(i)) \Rightarrow A[k] B[k]$.
• $\equiv$ (coincidence) = $df \preceq \cap$.
• $\prec$ (strictly precedence) is a strict form of $\preceq$ (precedence): $A$ StrictlyPrecedes $B$ (denoted $A \prec B$) if $(\forall i \in I_B) (k = idx_B(i)) \Rightarrow A[k] \prec B[k]$.
• $\sim$ (Alternates): $A$ Alternates $B$ (denoted $A \sim B$) if $(\forall i \in I_A) (k = idx_A(i)) \Rightarrow A[k] \preceq B[k] \prec A[k+1]$.

3.2 Interface Automata

We define interface automata to model the state transition of dynamic process.

Definition 3.3 [Interface automata] For a component $K$, an interface automaton $\varepsilon[K] = (S, s_0, \Sigma, E, C)$ is a finite state automaton where,
• $S$ is a finite set of states.
• $s_0 \in S$ is the initial state.
• $\Sigma$ is the alphabet set, where $\Sigma = \{ in, !out \ mid in \in K.Pp.M \land \text{out} \in K.Pr.M^* \}$. The event $in$ represents receiving provided service from reference interface and $!out$ is a finite sequence of services.
• $E \subseteq S \times \Sigma \times S \times 2^C \times \Phi(C)$ is the $\Sigma$-labeled transition relation. An edge $(s, a, s', \theta, \sigma)$ represents a transition from state $s$ to $s'$ on input symbol $a$. The set $\theta \subseteq C$ gives the clock to be reset with this transition, and $\sigma \in \Phi(C)$ is a clock constraint, and
• $C$ is a finite set of clocks.

4 Design of Automatic Train Control System

Communication Based Train Control (CBTC) is a train transportation control and protection system, which usually consists of Automatic Train Protection (ATP), Automatic Train Control (ATC) and Automatic Train Supervision (ATS) and ATP system is the core subsystem of CBTC. CBTC ensures the safe and accurate operation of trains on Automatic Train Control enabled lines [8].

4.1 System Overview

ATP ensures the safe and accurate operation of trains on Automatic Train Control enabled lines. ATP consists of the following devices (subsystems):
- ATP Processing Unit (PU) to receive train control commands from Trainlines and compare the train speed with limiting speed. It will also give braking command to Braking Equipment (BE) to apply brake.
- Trainlines, through which the control command and limiting speed are sent to ATP Processing Unit.
- Mode Direction Handler (MDH) for driver to select the drive mode, and each drive mode has a corresponding standard limiting speed.
- SpeedSensor to send actual speed of train to ATP PU.
- Train Operation Display (TOD) to display commands and speed information from Trainlines and other warning information from ATP PU.
- Train Information Management (TIM) for the management of train information such as drive mode and corresponding standard limiting speed.

4.2 Requirement Analysis

Model of UC1 (UC 1): Speed Supervision in RMF

The use case is modeled by the contract of the provided interface of a component: SuperviseSpeed, and the provided interface of a component: SpdSpvDeviceIF.

```java
class SpdSpvDevice implements SpdSpvDeviceIF:
   invariant ATP ≠ null \ ATP.speedSensor ≠ null \ ATP.modeList ≠ null
                   \ ATP.TOD ≠ null \ TIM ≠ null
   method initialTrain()
      pre: ATP.lights.findLight(mode) ≠ null
      post: mode' = 0 /*default drive mode: ATO*/
            ATP.lights.findLight(mode).turnOn() /*ATP light turns on*/
   method selectRMFMode()
      pre: ATP.lights.findLight(mode) ≠ null
      post: mode' = 3; /*RMF drive mode*/
            ATP.lights.findLight(mode).turnOn() /*RMF light turns on*/
   method acquireActualSpeed()
      pre: true
      post: aspd' = ATP.speedSensor.acquireSpeed()
   method acquireRestrictedSpeed(int mId, int rspd)
      pre: TIM.modeList.findMode(mId) ≠ null
      post: rspd' = TIM.modeList.findMode(mId).getRestrictedSpeed()
   method compareSpeed(int aspd, int rspd)
      pre: true
      post: dif' = rspd - aspd;
   method announceWarnMsg()
      pre: 0 < dif < criticalSpeed
      post: ATP.TOD.displayWarn()
   method brakeByHuman()
      pre: 0 < dif < criticalSpeed
      post: ATP.CBE.brakeByHuman()
   method emergencyBrake()
      pre: dif = 0
      post: ATP.EBE.brake()
```

772 Computers, Networks, Systems, and Industrial Applications
4.5 Dynamic Process Analysis

To ensure that the train is running in a safe condition, the real-time and safety properties of the system should be considered during design. **Informal description** The whole process of the brake by human and emergency brake with time constraint can be described as follows:

![Diagram of process analysis](image)

**Fig. 1.** Interface automata model

1. When the SpdController asks the PU to compute the difference between the limiting speed and current speed, the PU returns the computing result, \( \text{dif} \).
2. The SpdController informs the ATPSupport (here it acts like a connector) to display the warning message when \( \text{dif} \) is less than \( \text{criticalSpeed} \).
3. The ATPSupport requests the TOD to display the warning message to the train driver within 200ms. When the warn message is displayed in the TOD, the TOD will give a feedback message to the ATPSupport, which will also inform the SpdController that the warn message has displayed.
4. After the ATPSupport got the message from the SpdController (originally from the driver) to apply brake, it sends a brake request to BE and BE should finish the brake within 150 ms. Then the BE similarly gives a feedback message to the ATPSupport and the ATPSupport informs the SpdController that brake task has been finished.
5. If \( \text{dif} \) is less than or equal to zero, which means the current speed exceeds the limiting speed, the SpdController informs the ATPSupport to send the command of applying emergency brake to BE and BE should finish the brake within 100 ms. Then the BE similarly gives a feedback message to the ATPSupport and the ATPSupport informs the SpdController that emergency task has been finished.

**Interface automata Model** According to the description a for *Brake by Human* and *Emergency Brake*, we obtain corresponding interface automata models in Fig. 1.
5. Conclusion and Discussion

In this paper, we have presented an approach to model ATP System from a unified multiple views. Component based architecture is used to model networked Telemedicine System. We extended Timed Automata with multiply time characteristics by introducing clock and clock constraints. With this extension, we presented a component-based approach for the modeling of real-time system. A component-based system is a number of parallel components to realize an application by invoking services provided by other components. Interface Automata are defined for components interaction.

We will focus on the verification approach and tools base of interface automata model to ensure the correctness and safety of software system.

Acknowledgments. We would like to thank the reviewers of this paper for their valuable comments that help us to greatly improve the paper. This work is partially supported by the projects funded by the 973 program 2009CB320702, NSFC 61170084, NSFC for the Young of China 60903021, and ZuoHua Ding is supported by the NSF of China under Grant No.61170015, Zhejiang NSF under Grant No.Z1090357. The Lab is supported by NSFC Creative Team 61021004 and 863 Project 2011AA010101.

References